**Experiment 7**

**1. Write a Java program that reads a file name from the user, and then displays information about whether the file exists, whether the file is readable, whether the file is writable, the type of file and the length of the file in bytes**

import java.io.\*;

import java.util.\*;

class filedemo

{

public static void main(String args[])

{

File f=new File(args[0]);

System.out.println("name :"+f.getName());

System.out.println("path:"+f.getAbsolutePath());

System.out.println("exists:"+f.exists());

System.out.println("is file:"+f.isFile());

System.out.println("is dir:"+f.isDirectory());

System.out.println("read :"+f.canRead());

System.out.println("write:"+f.canWrite());

long l=f.lastModified();

Date d=new Date(l);

int date=d.getDate();

int month=d.getMonth();

int year=d.getYear();

int hh=d.getHours();

int mm=d.getMinutes();

int ss=d.getSeconds();

System.out.println(date+"/"+(month+1)+"/"+(1900+year));

System.out.println(hh+":"+mm+":"+ss);

}

}

**2. Write a Java program that reads a file and displays the file on the screen, with a line number before each line.**

import java.util.\*;

import java.io.\*;

class Rfile

{

public static void main(String args[])throws IOException

{

int j=1;

char ch;

Scanner scr=new Scanner(System.in);

System.out.print("\nEnter File name: ");

String str=scr.next();

FileInputStream f=new FileInputStream(str);

System.out.println("\nContents of the file are");

int n=f.available();

System.out.print(j+": ");

for(int i=0;i<n;i++)

{

ch=(char)f.read();

System.out.print(ch);

if(ch=='\n')

{

System.out.print(++j+": ");

}

}

}

}

**3. Write a Java program that displays the number of characters, lines and words in a text file.**

import java.util.\*;

import java.io.\*;

class Cfile

{

public static void main(String args[])throws IOException

{

int nl=1,nw=0;

char ch;

Scanner scr=new Scanner(System.in);

System.out.print("\nEnter File name: ");

String str=scr.nextLine();

FileInputStream f=new FileInputStream(str);

int n=f.available();

for(int i=0;i<n;i++)

{

ch=(char)f.read();

if(ch=='\n')

nl++;

else if(ch==' ')

nw++;

}

System.out.println("\nNumber of lines : "+nl);

System.out.println("\nNumber of words : "+(nl+nw));

System.out.println("\nNumber of characters : "+n);

}

}

**4. Write a Java program to illustrate collection classes like (i) Array List, (ii) Iterator, (iii)Hash map**

import java.util.\*;

class TestJavaCollection1{

list.add("Ravi");//Adding object in arraylist

list.add("Vijay");

list.add("Ravi");

list.add("Ajay");

Iterator itr=list.iterator();

while(itr.hasNext()){

System.out.println(itr.next());  }

}

}

**5. Convert the content of a given file into the uppercase content of the same file.**

import java.util.Scanner;

import java.io.\*;

public class UppercaseFileConverter2

{

public static void main(String[] args)throws IOException

{

String message;

String filename2;

Scanner keyboard = new Scanner(System.in);

System.out.print("Enter the filename: ");

filename = keyboard.nextLine();

FileWriter fwriter = new FileWriter(filename);

PrintWriter outputFile = new PrintWriter(fwriter);

System.out.println("Enter a message: ");

message = keyboard.nextLine();

outputFile.println(message);

outputFile.close();

System.out.println("Enter the name of the second file: ");

filename2 = keyboard.nextLine();

FileReader freader = new FileReader(filename2);

BufferedReader inputFile = new BufferedReader(freader);String str;

str = inputFile.readLine()

{

System.out.println(str);

String upper = message.toUpperCase();

str = inputFile.readLine(upper);

}

inputFile.close();

}

}

**Experiment 8**

1. **Write a java program that implements a multi-threaded application that has three threads. First thread generates a random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.**

import java.util.Random;

class Square extends Thread

{

int x;

Square(int n) {

x = n;

}

public void run()

{

int sqr = x \* x;

System.out.println("Square of " + x + " = " + sqr );

}

}

class Cube extends Thread

{

int x;

Cube(int n)

{

x = n;

}

public void run()

{

int cub = x \* x \* x;

System.out.println("Cube of " + x + " = " + cub );

}

}

class Number extends Thread {

public void run()

{

Random random = new Random();

for(int i =0; i<10; i++)

{

int randomInteger = random.nextInt(100);

System.out.println("Random Integer generated : " + randomInteger);

Square s = new Square(randomInteger);

s.start();

Cube c = new Cube(randomInteger);

c.start();

try {

Thread.sleep(1000);

} catch (InterruptedException ex) {

System.out.println(ex);

}

}

}

}

public class Java

{

public static void main(String args[])

{ Number n = new Number();

n.start();

}

}

**2.A program to illustrate the concept of multi-threading that creates three threads. First thread displays ―Good Morning‖ every one second, the second thread displays ―Hello‖ every two seconds and the third thread displays ―Welcome‖ every three seconds**

class GoodMorning extends Thread {

synchronized public void run() {

try {

int i=0;

while (i<5) {

sleep(1000);

System.out.println("Good morning ");

i++;

}

} catch (Exception e) {

}

}

}

class Hello extends Thread {

synchronized public void run() {

try {

int i=0;

while (i<5) {

sleep(2000);

System.out.println("hello");

i++;

}

} catch (Exception e) {

}

}

}

class Welcome extends Thread {

synchronized public void run() {

try {

int i=0;

while (i<5) {

sleep(3000);

System.out.println("welcome");

i++;

}

} catch (Exception e) {

}

}

}

class MultithreadDemo {

public static void main(String args[]) {

GoodMorning t1 = new GoodMorning();

Hello t2 = new Hello();

Welcome t3 = new Welcome();

t1.start();

t2.start();

t3.start();

}

}

**Experiment 9**

1. **Sorting using generic method**

public class SorterApp {

public static void main(String[] args) {

SorterApp app = new SorterApp();

app.run();

}

private void run() {

sort(new Integer[] { 1, 2, 3, 4, 5 });

sort(new Integer[] { 3, 1, 5, 4, 2 });

sort(new Integer[] { 5, 4, 3, 2, 1 });

System.out.println();

sort(new String[] { "a", "b", "c", "d", "e" });

sort(new String[] { "c", "a", "e", "d", "b" });

sort(new String[] { "e", "d", "c", "b", "a" });

}

private <T> extends Comparable<T> void sort(T[] values)

Sortable sorter = new BubbleSort();

sorter.sortAscending(values);

Utils.printArray(values);

}

}

**2.Stack using generic class**

class StackArray<T> implements Stack<T> {

private int maxSize;

private Object[] array;

private int top;

public StackArray(int maxSize) {

this.maxSize = maxSize;

this.array = new Object[maxSize];

this.top = -1;

}

...

public T pop() {

if(this.isEmpty())

throw new EmptyStackException();

return element(top--);

}

public T peek() {

if(this.isEmpty())

throw new EmptyStackException();

return element(top);

}

// Safe because push(T) is type checked.

@SuppressWarnings("unchecked")

private T element(int index)

{

return (T)array[index];

}

}

**3.Write a java program to find the maximum value from the given type of elements using a generic function**

public static <T extends Comparable<T>> T max(T... elements) {

T max = elements[0];

for (T element : elements) {

if (element.compareTo(max) > 0) {

max = element;

}

}

return max;

}

public static void main(String[] args)

{System.out.println("Integer Max: " + max(Integer.valueOf(32), Integer.valueOf(56), Integer.valueOf(89), Integer.valueOf(3), Integer.valueOf(456), Integer.valueOf(78), Integer.valueOf(45)));

System.out.println("Double Max: " + max(Double.valueOf(5.6), Double.valueOf(7.8), Double.valueOf(2.9), Double.valueOf(18.6), Double.valueOf(10.25), Double.valueOf(18.6001)));

System.out.println("String Max: " + max("Strawberry", "Mango", "Apple", "Pomegranate", "Guava", "Blackberry", "Cherry", "Orange", "Date"));

System.out.println("Boolean Max: " + max(Boolean.TRUE, Boolean.FALSE));

System.out.println("Byte Max: " + max(Byte.MIN\_VALUE, Byte.MAX\_VALUE));

}